Variational formulation and algorithm for trace operation in domain decomposition calculations

Jean-François Bourgat, Roland Glowinski, Patrick Le Tallec, Marina Vidrascu

▶ **To cite this version:**

Jean-François Bourgat, Roland Glowinski, Patrick Le Tallec, Marina Vidrascu. Variational formulation and algorithm for trace operation in domain decomposition calculations. [Research Report] RR-0804, INRIA. 1988. inria-00075747

HAL Id: inria-00075747
https://inria.hal.science/inria-00075747

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 804

VARIATIONAL FORMULATION AND ALGORITHM FOR TRACE OPERATOR IN DOMAIN DECOMPOSITION CALCULATIONS

Jean-François BOURGAT
Roland GLOWINSKI
Patrick LE TALLEC
Marina VIDRASCU

MARS 1988

RR804

VARIATIONAL FORMULATION AND ALGORITHM FOR TRACE OPERATOR IN DOMAIN DECOMPOSITION CALCULATIONS

Jean-François BOURGAT*, Roland GLOWINSKI**,
Patrick LE TALLEC***, Marina VIDRASCU*

* INRIA, Domaine de Voluceau, 78153 LE CHESNAY CEDEX, France.
** Dept. of Mathematics, University of Houston, 4800 Calhoun Road, HOUSTON, Texas 77004, USA.
*** Laboratoire Central des Ponts et Chaussées, 58, Bd. Lefèbvre, 75015 PARIS, France.

PAPIER RECUPERE ET RECYCLE

VATIATIONAL FORMULATION AND ALGORITHM FOR TRACE OPERATOR IN DOMAIN DECOMPOSITION CALCULATIONS

Abstract

A new preconditioning strategy is proposed for solving elliptic problems via domain decomposition techniques. This preconditioner acts on the Steklov- Poincare's operator (represented after discretization by the so-called Schur complement matrix) through the addition of a trace averaging and of the solution of a Neumann problem per subdomain. Such a strategy can operate on arbitrary geometries and unstructured meshes, gives the same role to each subdomain and can be written in a variational form. Two or three-dimensional numerical results are given to illustrate the efficiency of this strategy.

FORMULATION VARIATIONNELLE ET ALGORITHME POUR LES OPERATEURS DE TRACE INTERVENANT EN DECOMPOSITION DE DOMAINES

Résumé

On propose une nouvelle stratégie de préconditionnement pour la résolution de problèmes elliptiques par des techniques de décomposition de domaines. Ce préconditionneur agit sur l'opérateur de Steklov-Poincaré (représenté après discrétisation par la matrice complément de Schur) par un calcul de moyenne de traces et la résolution d'un problème de Neumann par sous-domaine. Cette stratégie peut opérer sur des géométries arbitraires, sur des maillages non structurés, fait jouer le même rôle à tous les sous-domaines et peut être écrite sous forme variationnelle. Elle est illustrée sur des exemples numériques bi ou tridimensionnels.

Mots-clés : Décomposition de domaines, opérateur de traces, gradient conjugué préconditionné.

1. Introduction. The idea of reducing the solution of an elliptic problem set on a domain Ω to the parallel solution of problems of same type set on subdomains Ω_i of Ω is ancient ([1],[2],[3]), but it gets new attention with the present development of parallel computers.

These domain decomposition methods are based on simple and intuitive ideas. Nevertheless, their numerical efficiency is very sensitive to the choice of the computational variables, that is to preconditioning. The approach proposed in this paper uses an H^1 norm on each subdomain (§3). On the product of the associated spaces, the problem takes a classical variational form and is associated to an operator which involves on each subdomain the successive solution of a Dirichlet and of a Neumann problem (§4). This formulation is then solved by a conjugate gradient algorithm (§5).

Numerical results will assert the efficiency of our approach (§6): on significative two dimensional situations (irregular mesh, discontinuous coefficients, crossed interfaces), the algorithm has converged in very few iterations. Similarly, convergence occured in 14 iterations in the solution of a Poisson problem set on a complex three-dimensional geometry.

It should be also observed that the present approach is the equivalent in a standard variational form (and using standard discretization techniques) of the mixed variational approach described in [4] within a mixed finite element framework.

2. A Simplified Model Problem. We first describe our approach on the following model problem:

$$\begin{cases} -\Delta u = f \text{ on } \Omega, \\ u = 0 \text{ on } \partial\Omega , \end{cases}$$

the domain Ω being decomposed as indicated in the figure below.

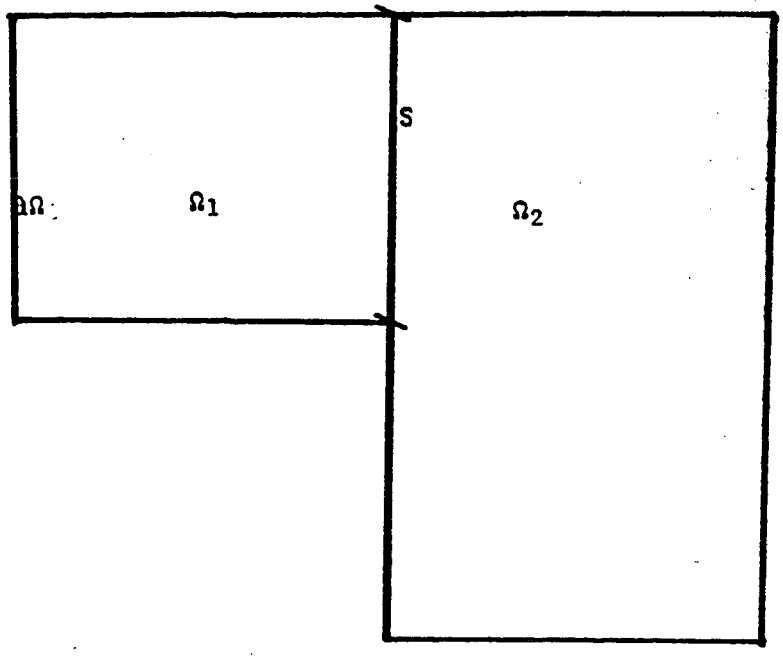

Figure 1: the model problem

Our goal is to solve the above problem only on the subdomains Ω_i. If we knew the value λ of the solution u on the interface S, then the parallel solution of

$$\begin{cases} -\Delta u_i = f \text{ on } \Omega_i, \\ u_i = \lambda \text{ on } S, \\ u_i = 0 \text{ on } \partial\Omega \cap \partial\Omega_i, \end{cases}$$

on Ω_1 and Ω_2 will achieve this goal. The problem is thus reduced to the computation of λ which can be done by the following gradient algorithm operating on the trace space $H_{00}^{1/2}(S)$:

<u>data:</u> λ given in $H_{00}^{1/2}(S)$;

<u>computation of the solution:</u> for any i, solve the Dirichlet problem:

$$\begin{cases} -\Delta u_i = f \text{ on } \Omega_i, \\ u_i = \lambda \text{ on } S, \\ u_i = 0 \text{ on } \partial\Omega \cap \partial\Omega_i; \end{cases}$$

<u>computation of the gradient:</u> for any i, solve the Neumann problem (preconditioner)

$$\begin{cases} -\Delta\psi_i = 0 \text{ on } \Omega_i , \\ \dfrac{\partial\psi_i}{\partial n_i} = \dfrac{1}{2}\left(\dfrac{\partial u_1}{\partial n_1} + \dfrac{\partial u_2}{\partial n_2}\right) \text{ on } S, \\ \psi_i = 0 \text{ on } \partial\Omega \cap \partial\Omega_i ; \end{cases}$$

<u>updating:</u> set $\lambda = \lambda - \rho(\psi_1 + \psi_2)$ and reiterate .

The domain decomposition method that we will now introduce is simply a generalization of this algorithm.

3. The Original Problem. Consider the domain Ω, partitioned into subdomains Ω_i as indicated on Figure 2.

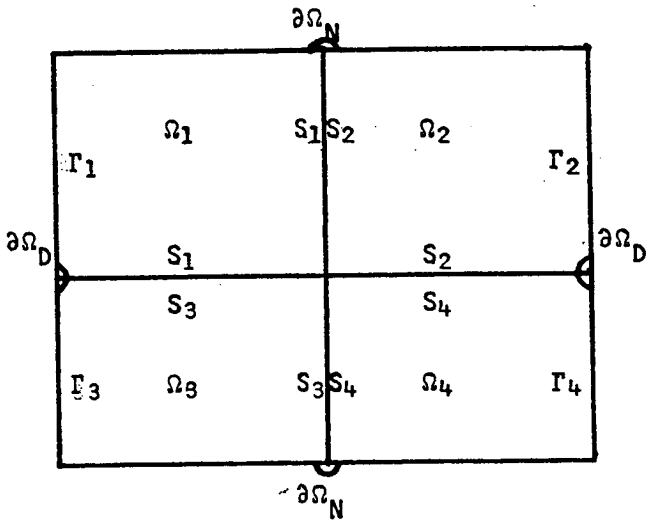

Figure 2: definition of the subdomains and boundaries.

Let us introduce the boundaries (see Figure 2)

$$\partial\Omega = \partial\Omega_N \cup \partial\Omega_D ,$$

$$\Gamma_i = \partial\Omega_D \cap \partial\Omega_i ,$$

$$S_i = \partial\Omega_i - \Gamma_i - \text{interior}(\partial\Omega_N \cap \partial\Omega_i),$$

together with the spaces

$$V = \left\{ v \in H^1(\Omega\ ; \mathbb{R}^p),\ v = 0 \text{ on } \partial\Omega_D \right\},$$

$$V_i = \left\{ v \in H^1(\Omega_i\ ; \mathbb{R}^p),\ v = 0 \text{ on } \Gamma_i \right\},$$

$$V_{0\,i} = \left\{ v \in H^1(\Omega_i\ ; \mathbb{R}^p),\ v = 0 \text{ on } \Gamma_i \cup S_i \right\}.$$

As in [1], the pairing $\tilde{a}_i(u,v)$ will denote a given scalar product on V_i, X will be the product of the spaces V_i and Y will represent the space of traces on S of functions of V. In addition, $Tr_i^{-1}(\lambda)$ will represent any element z of V_i whose trace on S_i is equal to λ. Finally, we introduce the elliptic form

$$a_i(u,v) = \int_{\Omega_i} A_{mnkl}(x) \frac{\partial u_m}{\partial x_n} \frac{\partial v_k}{\partial x_l}.$$

Under these notations, the problem to solve writes

$$\begin{array}{l} \underline{\text{Find}}\ u \in V\ \underline{\text{such that}} \\ \sum_i a_i(u,v) = \langle f, v \rangle,\ \forall\ v \in V\ . \end{array}$$

4. Domain Decomposition of the Original Problem.

4.1 Introduction of an operator of Steklov-Poincaré's type.

We first define a trace operator α_i from V_i into Y satisfying

$$\text{(C)} \qquad \sum_i \alpha_i(v) = Tr(v),\ \forall\ v \in V.$$

For example, at the continuous level, we can set $\alpha_i(v) = Tr(v)/2$. A different definition should be used at the finite element level in order for condition (C) to be still satisfied after discretization. A possible choice will be described in §6.

For $w = (w_i) \in X = \prod_i V_i$, we then set

$$\lambda = \sum_i \alpha_i(w_i),$$

solve the Dirichlet problems

$$\begin{cases} a_i(z_i, v) = 0 \ , \ \forall\ v \in V_{0i} \ , \\ z_i \in V_i \ , \ z_i' = \lambda \text{ on } S_i \ , \end{cases}$$

define the usual Steklov-Poincare's operator by

$$< S\lambda \ , \ \lambda' > = \sum_j a_j(z_j, Tr_j^{-1}(\lambda')) \ , \quad \forall\ \lambda' \in Y,$$

and solve the Neumann problems

$$\begin{cases} \tilde{a}_i(\psi_i, v) = < S\lambda \ , \ \alpha_i v > \ , \ \forall\ v \in V_i \ , \\ \psi_i \in V_i \ . \end{cases}$$

Our final operator is now defined from X into itself by

$$\mathcal{A}(w) = \{\psi_i\} \ .$$

Remark 4.1: In computing $a_j(z_j, Tr_j^{-1}(\lambda'))$, the choice of the representative element of Tr_j^{-1} is of no importance since, by construction, z_j is orthogonal to any component of this element in $Ker(Tr_j)$.

4.2 Analysis of $\mathcal{A}$. Assuming a_i to be symmetric positive on V_i, we introduce the operator K_i defined on V_i by

$$\tilde{a}_i(K_i u, K_i v) = a_i(u, v) \ , \ \forall\ u, v \in V_i \ ,$$

and the operator $\mathcal{B}$ defined on X by $\mathcal{B}\{w_i\} = \{K_i z_i\}$. By construction, we have:

$$\begin{aligned} (\mathcal{A}(w), w') &= \sum_i \tilde{a}_i(\psi_i, w_i') = \sum_i < S\lambda \ , \ \alpha_i w_i' > \\ &= < S\lambda \ , \ \lambda' > = \sum_j a_j(z_j, Tr_j^{-1}(\lambda')) \\ &= \sum_j a_j(z_j, z_j') = (\mathcal{B}(w), \mathcal{B}(w')) . \end{aligned}$$

Therefore, $\mathcal{A}$ is a self-adjoint positive operator from X into itself, and can be decomposed into the product $\mathcal{A} = \mathcal{B}^T\mathcal{B}$. In particular, standard conjugate gradient algorithms operating on $\mathcal{A}$ will converge in $Im(\mathcal{B}^T)$ with rate

$$\frac{1 - cond(\mathcal{B})}{1 + cond(\mathcal{B})} \ .$$

Moreover, from the above relations, we have

$$(\mathcal{A}w,w') = < S\lambda , \lambda' >.$$

In other words, $\mathcal{A}$ is a preconditioned version of the Steklov-Poincare's operator S. Such operators have been extensively studied in [5], [6], [7], [8]. Compared to them, our operator $\mathcal{A}$ adds a Neumann problem associated to the boundary condition

$$\frac{\partial \psi_i}{\partial n_i} = \alpha_i \sum_j \frac{\partial z_j}{\partial n_j},$$

which sends the dual of X back in X. The introduction of this preconditioner together with the averaging α_i is the main originality of the present approach.

Remark 4.2: The degrees of freedom in the definition of the operator $\mathcal{A}$ are the choice of the trace averaging α_i and of the scalar product $\tilde{a}_i(u,v)$. The choice of α_i has already been discussed. As for the scalar product, the best possible choice would be:

$$\tilde{a}_i(u,v) = a_i(u,v)$$

(K_i = Id), because then the Neumann step corresponds to the exact inverse of the Steklov-Poincaré's operator introduced in the Dirichlet part. With this choice we can expect $\mathcal{A}$ to have a very small condition number. Unfortunately, if Γ_i is empty, $a_i(u,v)$ does not define a scalar product on V_i. In that situation, the best choice is then

$$\tilde{a}_i(u,v) = a_i(u,v) + \sum_{k=1}^{K} <u,v_{ik}> <v,v_{ik}>$$

with $\{v_{ik}\}$ a sequence of elements of V_i adequately chosen. At the discrete level, this means adding positive diagonal terms to the matrix associated to $a_i(u,v)$.

4.3 Variational writing in X. To define the right hand side of our new problem, we solve the Dirichlet problems

$$\begin{cases} a_i(z_i^0,v) = <f,v> , \ \forall\ v \in V_{0i}, \\ \\ z_i^0 \in V_{0i}, \end{cases}$$

and the Neumann problems

$$\begin{cases} \tilde{a}_i(\psi_i^0, v) = \sum_j a_j(z_i^0, Tr_j^{-1}(\alpha_i v)) - \langle f, Tr_j^{-1}(\alpha_i v)\rangle, \quad \forall\, v \in V_i\ , \\ \\ \psi_i^0 \in V_i\ . \end{cases}$$

In that framework, the problem to solve can be written under the equivalent form

(P) $\quad$ Solve $\mathcal{A}(w) = -\{\psi_i^0\}$ in $\mathrm{Im}(\mathcal{B}^T)$,

within the identification $u = z_i + z_i^0$ on Ω_i .

4.4 Equivalence proof. Let w be the solution of problem (P), solution which uniquely exists from Lax-Milgram's lemma. Let u be the field given by

$$u = z_i + z_i^0 \text{ on } \Omega_i\ .$$

By construction the traces of z_i and z_i^0 are compatible on the interfaces S_i and are equal to zero on the boundaries Γ_i . Thus u belongs to V.

Now, let v be any element of V. On each subdomain, v can be decomposed into

$$v = v_j + Tr_j^{-1}(Tr\ v) \text{ with } v_j = v - Tr_j^{-1}(Tr\ v) \in V_{0j}\ .$$

Therefore, by construction of u, z_i and z_i^0, we have

$$\begin{aligned} \sum_j a_j(u,v) &= \sum_j a_j(z_j + z_j^0,\ v_j + Tr_j^{-1}(Tr\ v)) \\ &= \sum_j \langle f, v_j\rangle + \sum_j a_j(z_j + z_j^0,\ Tr_j^{-1}(Tr\ v)) \\ &= \sum_j \langle f, v_j\rangle + \sum_j a_j(z_j + z_j^0,\ Tr_j^{-1}(\textstyle\sum_i \alpha_i v)) \\ &= \sum_j \langle f, v_j\rangle + \sum_i\sum_j a_j(z_j + z_j^0,\ Tr_j^{-1}(\alpha_i v)) \\ &= \sum_j \langle f, v_j\rangle + \sum_i \tilde{a}_i(\psi_i + \psi_i^0,\ v) \\ &\quad + \sum_i\sum_j \langle f,\ Tr_j^{-1}(\alpha_i v)\rangle \end{aligned}$$

$$= \sum_j \langle f, v_j + Tr_j^{-1}(Tr\ v)\rangle + \sum_i \tilde{a}_i(\psi_i + \psi_i^0,\ v).$$

But, by construction of w, we have $\psi_i = -\psi_i^0$ and thus we finally get

$$\sum_j a_j(u,v) = \sum_j \langle f, v_j + Tr_j^{-1}(Tr\ v)\rangle = \langle f, v\rangle \ ,\ \forall v \in V.$$

In other words, the solution u that we have constructed is the unique solution of our original problem.

5. Solution Algorithm. The solution algorithm that we propose is a standard conjugate gradient method ([9]) applied to the inversion of $\mathcal{A}$ on $Im(\mathcal{B}^T)$. It writes

INITIALIZATION

For λ given in Y, solve the Dirichlet problems:

$$\begin{cases} a_i(u_{i,0}, v) = \langle f, v\rangle, \ \forall\ v \in V_{0i}, \\ u_{i,0} = \lambda \text{ on } S_i, \\ u_{i,0} \in V_i ; \end{cases}$$

compute $L_i(v) = \sum_j \left(a_j(u_j, Tr_j^{-1}(\alpha_i v)) - \langle f, Tr_j^{-1}(\alpha_i v)\rangle\right)$,

$\forall\ v \in V_i$;

solve the Neumann problems:

$$\begin{cases} \tilde{a}_i(\varphi_{i,0},\ v) = L_i(v), \ \forall\ v \in V_i, \\ \varphi_{i,0} \in V_i \ ; \end{cases}$$

compute $d_0 = \sum_i \tilde{a}_i(\varphi_{i,0}, \varphi_{i,0}) = \sum_i L_i(\varphi_{i,0})$;

set $w_{i,0} = \varphi_{i,0}$ and $R_{i,0}(v) = L_i(v)$.

LOOP ON n: for n =0 until satisfied do

computation of $\mathcal{A}w$

compute $\lambda = \sum_j \alpha_j\ w_{j,n}$;

solve the Dirichlet problems:

$$\begin{cases} a_i(z_i, v) = 0, \ \forall\, v \in V_{0i}, \\ z_i = \lambda \text{ on } S_i, \\ z_i \in V_i; \end{cases}$$

compute $L_i(v) = \sum_j \left(a_j(z_j, Tr_j^{-1}(\alpha_i v)) \right)$, $\forall\, v \in V_i$;

solve the Neumann problems:

$$\begin{cases} \tilde{a}_i(\psi_i, v) = L_i(v), \ \forall\, v \in V_i, \\ \psi_i \in V_i. \end{cases}$$

descent

set $r_n = \sum_j \tilde{a}_j(w_{j,n}, \psi_j) = \sum_j L_j(w_{j,n})$

set $\rho_n = d_n / r_n$;

set:

$$\begin{cases} u_{i,n+1} = u_{i,n} - \rho_n z_i, \\ \varphi_{i,n+1} = \varphi_{i,n} - \rho_n \psi_i, \\ R_{i,n+1}(v) = R_{i,n}(v) - \rho_n L_i(v). \end{cases}$$

computation of the new descent direction

compute $d_{n+1} = \sum_j \tilde{a}_j(\varphi_{j,n+1}, \varphi_{j,n+1}) = \sum_j R_{j,n+1}(\varphi_{j,n+1})$

or alternatively $d_{n+1} = d_n - \rho_n \sum_j L_j(\varphi_{j,n} + \varphi_{j,n+1})$;

stop if $\sqrt{d_{n+1}/d_0} < 10^{-5}$;

set $w_{i,n+1} = \varphi_{i,n+1} + \dfrac{d_{n+1}}{d_n} w_{i,n}$.

Remark 5.1: Observe that the linear forms $R_i(v)$ and $L_i(v)$ only operate on the traces on S_i of the functions v. This simplifies their computation. Moreover, this means that, in the algorithm, we just have to compute and store the traces on S_i of the functions φ_i and w_i.

Remark 5.2: The above algorithm is modular and, since it does not involve any relaxation parameter, is completely transparent to the user. Moreover, it has a high degree of parallelism: most of the CPU is devoted to the solution on each subdomain of independent Dirichlet and Neumann problems. The only "rendez-vous" are the trace averaging operations which occur twice per iteration.

6. Numerical Results. Our numerical results deal with the Laplace operator for which we have

$$a_i(u,v) = \int_{\Omega_i} \beta_i \ \nabla u.\nabla v \ .$$

The spaces V_i are approximated by finite elements of P1-Lagrange type. The trace operators α_i are defined at each node of the interface by the formula

$$(\alpha_i(v))(M_k) = \left(a_i(\varphi_k,\varphi_k) / (\sum_j a_j(\varphi_k,\varphi_k)) \right) v(M_k)$$

where φ_k denotes the weighting function associated to the node M_k. The three-dimensional coding was done within the MODULEF finite element library in a multielement, multiproblem and multitasking framework.

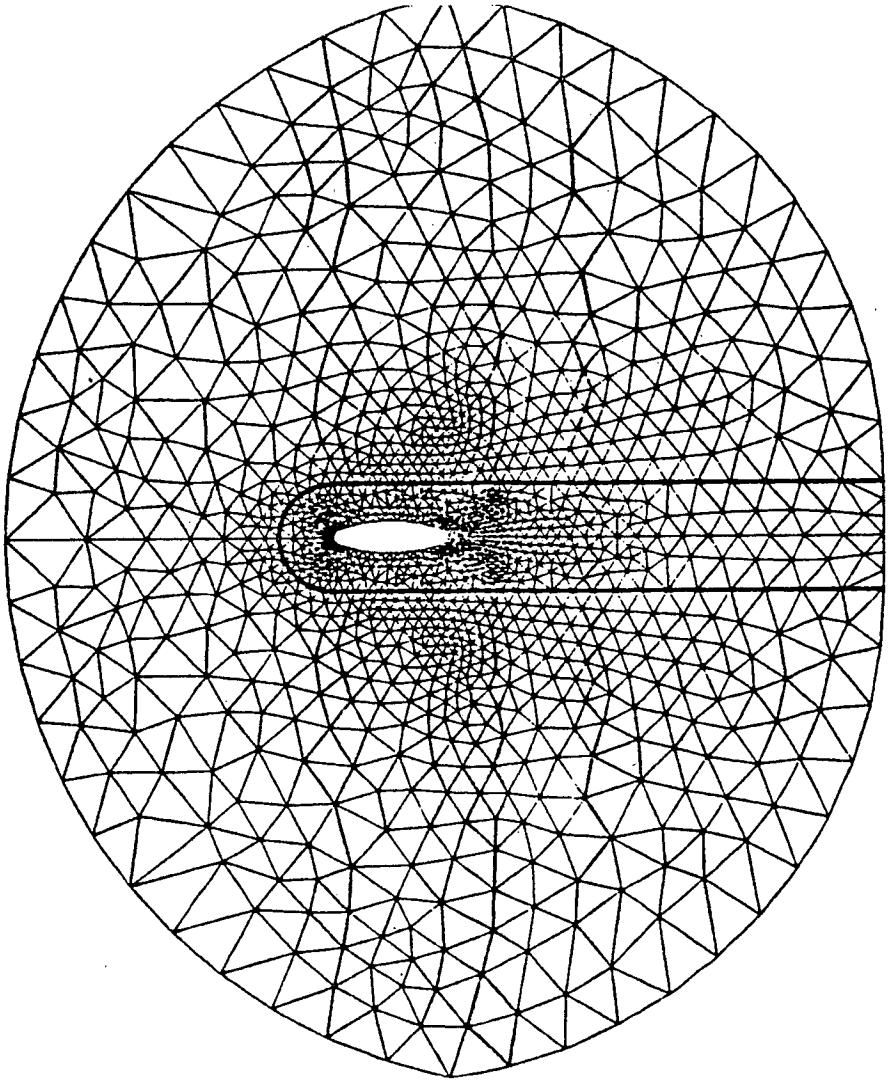

Figure 3

The first computed geometry is described on Figure 3. The internal mesh (on Ω_1) had 1222 triangles and 669 vertices, the external mesh (on Ω_2) had 1440 triangles and 780 vertices. For the pure Dirichlet problem, the algorithm has converged in 4 iterations for $\beta_1=\beta_2=1$ and in 3 iterations for $\beta_1=0.1$, $\beta_2=1$.

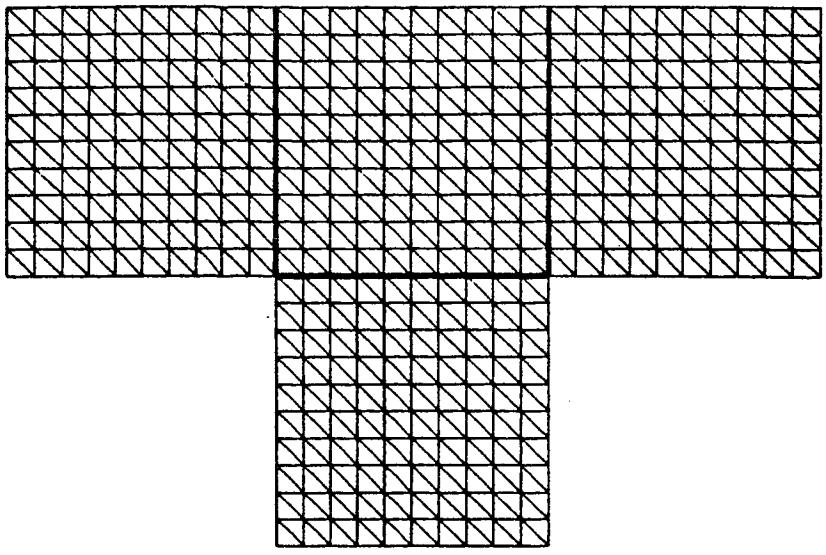

Figure 4: the T shaped domain

The second test considers a pure Dirichlet problem set on the T-shaped domain treated in [10]. On this example, the algorithm has converged in 2 iterations.

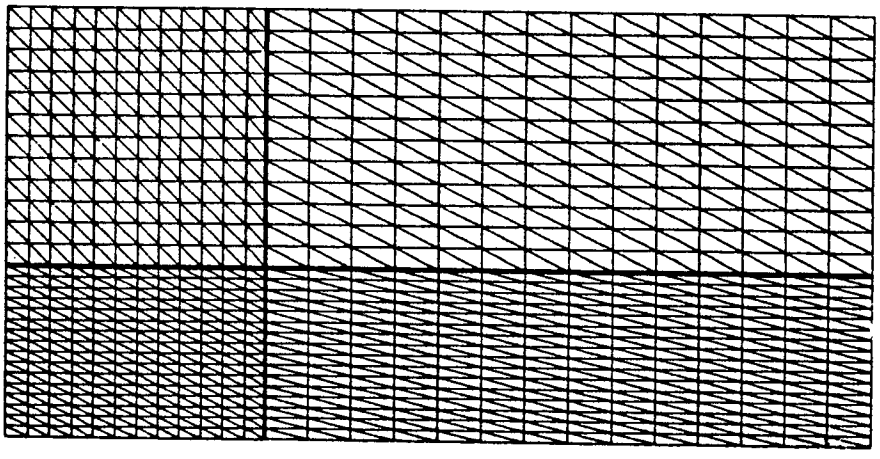

Figure 5: the "square cross" rectangle

The domain for the third example is the rectangle $\Omega=]0,4[\times]0,2[$ which is divided in 4 subdomains with crossing interfaces, as indicated on Figure 5. For both choices $\beta_1=\beta_2=\beta_3=\beta_4=1$ and $\beta_1=1.,\beta_2=10.,\beta_3=0.1,\beta_4=0.01$, the algorithm has converged in 5 iterations in the case of pure Dirichlet boundary conditions.

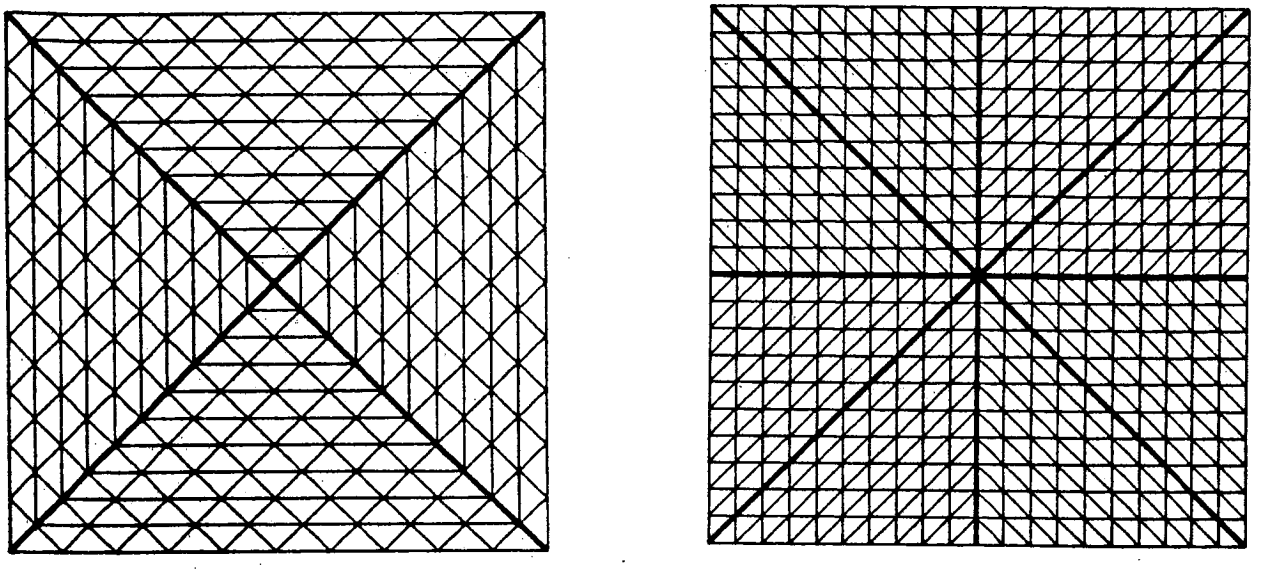

Figure 6: the "oblique cross" rectangle

The fourth domain that we have treated is the unit square divided as indicated in Figure 6. Here both the pure Dirichlet and the pure Neumann problem were considered, with $\beta_i = 10^{-3}$ if $x_1 - x_2 < 0$ and $\beta_i = 1$ if not . For the Dirichlet (resp. Neumann) problem, convergence was reached after 3 (resp. 8) iterations in the case of 4 subdomains and after 6 (resp. 20) iterations in the case of 8 subdomains. Taking 200 triangles per subdomain instead of 100 did not change the number of required iterations.

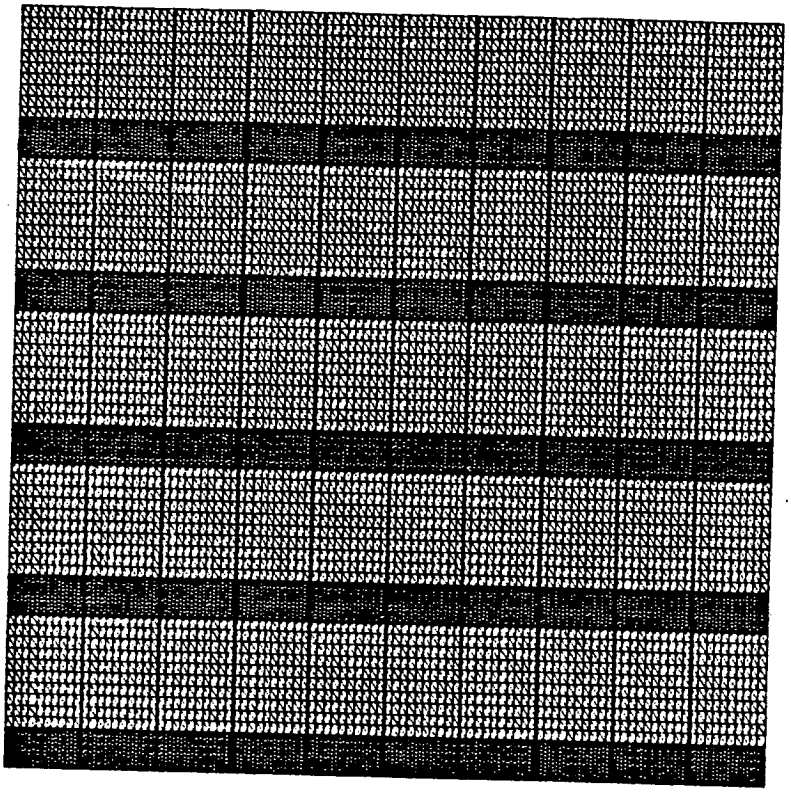

Figure 7: the "checkaboard" domain

The next domain that we have treated has 100 subdomains, 20000 triangles and 10201 nodes. On this domain, represented in Figure 7, we have treated both the pure Dirichlet and the pure Neumann problem. For β_i=1 everywhere convergence was reached after 40 iterations for the Dirichlet problem and after 74 iterations for the Neumann problem. For $\beta_i = 1$ in the wide strips and $\beta_i = 10^{-3}$ elsewhere, the Dirichlet (resp. Neumann) problem did require 54 (resp. 173) iterations to converge. Observe that there were 1701 (resp. 1737) nodes on the interface for the Dirichlet (resp. Neumann) problem.

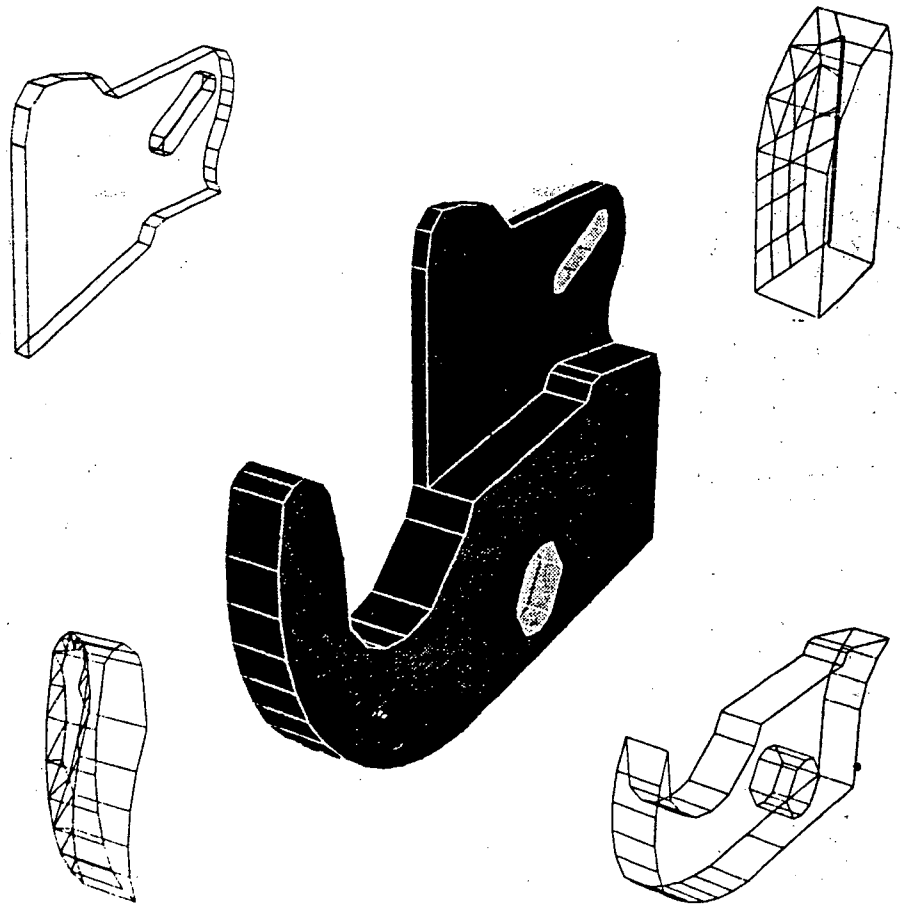

Figure 8: three-dimensional geometry

Our last numerical example deals with the complex three-dimensional geometry of Figure 8. The domain here is divided in 4, contains 937 nodes, with 83 nodes at the interfaces. Dirichlet boundary conditions were imposed only on the internal hole. Convergence occured after 14 iterations.

On all our numerical tests, we have observed that the number of iterations before convergence was independent of:
(i) the initial value of the trace λ,
(ii) the values of the right hand-side,
(iii) the discretization step.

7. Conclusion. In conclusion, all these numerical tests assess the validity of the conjugate gradient algorithm when operating on the product of traces, at least within the framework of a scalar elliptic operator and of a moderate number of subdomains. Corners in the decomposition can be handled easily provided that the averaging trace α_i be properly defined and be present in the Neumann step.

REFERENCES

(1) J.H. BRAMBLE, J.E. PASCIAK et A.H. SCHATZ, The construction of preconditioners for elliptic problems by substructuring.I, Math. Comp. 47 (1986), pp. 103-134.
(2) O. WIDLUND, Iterative methods for elliptic problems on regions partitionned into substructures and the biharmonic Dirichlet problem, Proceedings of the 6th International Conference on Computing Methods In Applied Science and Engineering, Versailles, North-Holland, 1983.
(3) R.GLOWINSKI, G.H. GOLUB, J. PERIAUX, eds.,Proceedings of the 1rst international symposium on decomposition methods for partial differential equations, Paris, SIAM editor, Philadelphia 1988.
(4) R. GLOWINSKI, M. WHEELER, Domain decomposition methods for mixed finite element approximation, in (3).
(5) V.I. AGOSHKOV, Poincare-Steklov's operators and domain decomposition methods in finite dimensional spaces, in (3).
(6) V.I. LEBEDEV, V.I. AGOSHKOV, The variational algorithms of domain decomposition method, Preprint n°54, Department of Numerical Mathematics of the Academy of Sciences, URSS, Moscow, 1983. (In Russian).
(7) V.I. AGOSHKOV, V.I. LEBEDEV, The Poincare-Steklov's operators and the domain decomposition methods in variational problems, in Computational processes and systems, NAUKA, Moscow (1985), pp. 173-227. (In Rusian.)
(8) V.I. LEBEDEV, The decomposition method, Department of Numerical Mathematics of the Academy of Sciences, URSS, Moscow, 1986. (In Russian.)
(9) R. GLOWINSKI, Numerical methods for nonlinear variational problems, Springer Verlag, 1984.
(10) L.D. MARINI, A. QUARTERONI, An iterative procedure for domain decomposition methods: a finite element approach, in (3).

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique